REMARKS

Applicant respectfully requests reconsideration and allowance of the subject application. Claims 1-40 are pending in this application.

In the December 6 Office Action, the Examiner requested that any response be accompanied by a 3½ inch IBM format floppy disk containing a duplicate copy of the response. In accordance with this request, such a floppy disk accompanies this response.

Applicant thanks the Examiner for the telephonic interview of October 4, 2001, at which the pending claims and the Matsumoto (U.S. Patent No. 5,835,765) and Kubo (U.S. Patent No. 5,881,284) references were discussed but no agreement on the allowability of any of the claims was reached.

Claims 1-39 stand rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 5,835,765 to Matsumoto (hereinafter "Matsumoto") in view of U.S. Patent No. 5,881,284 to Kubo (hereinafter "Kubo"). Applicant respectfully submits that claims 1-39 are not obvious over Matsumoto in view of Kubo.

Matsumoto is directed to a computer operation management system for a computer operating system capable of simultaneously executing multiple application programs (see, Abstract). The underlying operating system in Matsumoto has a virtual storage area with a finite size – when a program is executed exceeding this finite size the program ends immediately (see, col. 9, lines 5-7). The management system of Matsumoto is situated between the operating system and the application programs, and operates to monitor and control execution of the application programs (see, Fig. 1 and col. 8, lines 50-60). Rather than having the operating system terminate an application, the management

system of Matsumoto monitors resource usage and initiates its own recovery process if a threshold is exceeded (see, col. 19, lines 52-57; and col. 16, lines 29-32).

The management system of Matsumoto includes a resource manager that checks an amount of memory used and then compares the actual memory used with a control limit read from a program definition file (see, col. 16, lines 29-32). If the control limit is exceeded, an error recovery processor is notified (see, col. 16, lines 32-35). The error recovery process executes a predefined error recovery process also stored in the program definition file (see, col. 10, lines 37-40). The error recovery process comprises what measures to implement when an error occurs, the recovery procedure, alarm notification, mail, and what measures to implement for any related programs (see, col. 15, lines 10-14).

Kubo is directed to scheduling a job in a clustered computer system (see, Abstract). In Kubo, a job selector selects jobs for execution by one of multiple different clusters (see, col. 3, lines 8-18). A measurement mechanism exists in each cluster to measure the resource utilization in the cluster after the previous measurement (see, col. 3, lines 22-26). A request can be sent to the job selector to start scheduling a new job for the cluster at two different times: one is when the measurement is completed (see, Fig. 3), and the other is when a job is completed (see, Fig. 4). In the first situation, when the measurement is completed, if the resource utilization is low then the request controller requests the job selector to start a new job in the cluster; however, if the resource utilization is not low, then a new job is not requested (see, col. 3, lines 41-52). In the second situation, when execution of a job is completed, the request controller judges whether the resource utilization of the cluster that completes execution is high (see, col. 3, line 65 – col.

4, line 3). If the utilization in the cluster is not high, then a request is sent to the job selector to start scheduling a new job for the cluster; however, if the utilization in the cluster is high, then such job selection is not required (see, col. 4, lines 4-9). Thus, although Kubo teaches two different trigger points (utilization high and utilization low) for determining whether to request a job be scheduled, which of these trigger points is used is based on whether a resource utilization measurement was just completed or execution of a job was just completed.

Additionally, when execution of a job is completed, the request controller judges whether the resource utilization is high based on a first threshold value that is set to a value near a target resource utilization rate and a second threshold value is set to a value larger than the first threshold value (see, col. 4, lines 16-20). If the measured value exceeds the second threshold value, then the resource utilization is judged to be high, and if not, then the utilization is judged not to be high (see, col. 4, lines 20-23).

To establish a *prima facie* case of obviousness, three basic criteria must be met. First, there must be some suggestion or motivation, either in the references themselves or in the knowledge generally available to one of ordinary skill in the art, to modify the reference or to combine reference teachings. Second, there must be a reasonable expectation of success. Finally, the prior art reference (or references when combined) must teach or suggest all the claim limitations. **The teaching or suggestion to make the claimed combination and the reasonable expectation of success must both be found in the prior art, and not based on applicant's disclosure**. See, emphasis added, MPEP § 2142.

Applicant respectfully submits that it would not have been obvious to one of ordinary skill in the art to combine Matsumoto and Kubo. As discussed above,

Matsumoto is directed to monitoring execution of application programs on a computer and invoking an error recovery process if an amount of memory used exceeds a control limit. Kubo, on the other hand, is directed to scheduling execution jobs across multiple different processor clusters. Applicant respectfully submits that there is no teaching or suggestion to combine the monitoring and invoking of an error recovery process of Matsumoto with the job scheduling of Kubo.

However, assuming for the sake of argument that the Matsumoto and Kubo references are combined, Applicant respectfully submits that the combination does not disclose or suggest a method of controlling memory usage as recited in claim 1. Claim 1 recites:

> setting a plurality of memory thresholds; and
> the operating system wielding, at increasingly critical memory thresholds, correspondingly increasing control over said one or more application programs to reduce memory usage.

Kubo simply discloses determining whether a cluster is ready to have a new job scheduled for it. Which of two different resource utilization levels are used to make the determination are based on the event that caused the determination to occur. One such event is the resource utilization measurement that occurs at predetermined times (see, col. 3, lines 22-26 and 41-52). The other such event is job execution completion (see, col. 3, line 65 – col. 4, line 3). Applicant respectfully submits that these two events are not increasingly critical memory thresholds. There is no discussion or suggestion in Kubo of an "increasingly critical" nature to these events – they are simply two different events that can be used to determine whether a cluster is ready to have a new job scheduled for it.

Kubo also discloses that, in determining whether a cluster is ready to have a new job scheduled for it due to completing a previous job execution, the determining is based on whether its second threshold is exceeded (see, col. 4, lines 20-23). This decision is based on the second threshold, not on the first threshold (the first threshold appears to simply be a reference point for determining what the second threshold should be). Whether the measured resource utilization exceeds or does not exceed the first threshold is irrelevant – Kubo discloses no comparison of any values to this first threshold in order to determine whether job scheduling is needed. Thus, there is no concept in Kubo of increasing control at increasingly critical thresholds as claimed in claim 1. There are only two possible results in Kubo when completing a previous job execution – a job is selected for execution by the cluster or a job is not selected for execution by the cluster. Which of these results occurs is based solely on the second threshold – there is nothing about increasing the control over the cluster for the different thresholds.

Thus, Applicant respectfully submits that Kubo does not disclose or suggest the operating system wielding, at increasingly critical memory thresholds, correspondingly increasing control over said one or more application programs to reduce memory usage as claimed in claim 1.

Furthermore, Applicant respectfully submits that Matsumoto does not cure this deficiency of Kubo. In Matsumoto, a computer operation management system including a computer resource manager exists between the operating system and the applications (see, Figs. 1 and 2). This allows the management system to monitor resource usage and initiate its own recovery process if the threshold is exceeded, rather than having the operating system immediately terminate the application and generate an error (see, col. 19, lines 52-57; and col. 16, lines 29-

32). Thus, notification of the error recovery means is the only action taken because the management system is able to intervene before the operating system would have to terminate (abnormally end) the application – the abnormal endings are thus prevented (see, col. 19, lines 52-57).

Thus, Applicant respectfully submits that Matsumoto does not disclose the operating system wielding, at increasingly critical memory thresholds, correspondingly increasing control over said one or more application programs to reduce memory usage as claimed in claim 1. Matsumoto's goal is to replace one level of control (termination) with another level of control (notice), not to have the two levels coexist. Furthermore, it is two different entities that are providing the level of control – the management system provides the notice level while the underlying operating system provides the termination level. Applicant respectfully submits that there is no disclosure or suggestion in Matsumoto of a single entity providing both of these levels, and thus no disclosure or suggestion of an operating system wielding increasing control as claimed in claim 1.

Thus, for at least these reasons, Applicant respectfully submits that claim 1 is allowable over Matsumoto in view of Kubo.

With respect to claim 2, it was asserted in the June 6 Office Action that "limiting, closing, or terminating of a program are well known in the art (MPEP2144.03)" (see, ¶4, p. 3). Applicant respectfully traverses this rejection and submits that "communicating a request to at least one of the application programs for the at least one **application program to limit its use of memory**" as claimed in claim 2 is not well known in the art. Applicant respectfully requests that the rejection to claim 2 be withdrawn or evidence cited teaching "communicating a

request to at least one of the application programs for the at least one application program to limit its use of memory" as claimed in claim 2.

With respect to claim 3, it was asserted in the June 6 Office Action that "limiting, closing, or terminating of a program are well known in the art (MPEP2144.03)" (see, ¶4, p. 3). Applicant respectfully traverses this rejection and submits that wielding increasing operating system control comprising "prompting a user to select at least one of the application programs and then the operating system requesting that the at least one selected application program close itself" as claimed in claim 3 is not well known in the art. Applicant respectfully requests that the rejection to claim 3 be withdrawn or evidence cited teaching this element.

With respect to claim 4, it was asserted in the June 6 Office Action that "limiting, closing, or terminating of a program are well known in the art (MPEP2144.03)" (see, ¶4, p. 3). Applicant respectfully traverses this rejection and submits that wielding increasing operating system control comprising "prompting a user to select at least one of the application programs and then terminating it without allowing its further execution" as claimed in claim 4 is not well known in the art. Applicant respectfully requests that the rejection to claim 4 be withdrawn or evidence cited teaching this element.

With respect to claim 5, it was asserted in the June 6 Office Action that "limiting, closing, or terminating of a program are well known in the art (MPEP2144.03)" (see, ¶4, p. 3). Applicant respectfully traverses this rejection and submits that wielding increasing operating system control comprising:

> at a first memory threshold, requesting at least one of the application programs to limit its use of memory;
>
> at a second memory threshold, requesting at least one of the application programs to close itself; and

at a third memory threshold, terminating at least one of the application programs without allowing its further execution.

as claimed in claim 5 is not well known in the art. Applicant respectfully requests that the rejection to claim 5 be withdrawn or evidence cited teaching these elements.

With respect to claim 6, it was asserted in the June 6 Office Action that "limiting, closing, or terminating of a program are well known in the art (MPEP2144.03)" (see, ¶4, p. 3). Applicant respectfully traverses this rejection and submits that wielding increasing operating system control comprising:

> at a first memory threshold, requesting at least one of the application programs to limit its use of memory;
> at a second memory threshold, prompting a user to select at least one of the application programs and then requesting it to close itself; and
> at a third memory threshold, prompting the user to select at least one of the application programs and then terminating it without allowing its further execution.

as claimed in claim 6 is not well known in the art. Applicant respectfully requests that the rejection to claim 6 be withdrawn or evidence cited teaching these elements.

With respect to claims 7-9, claims 7-9 depend from claim 1 and Applicant respectfully submits that claims 7-9 are likewise allowable over the cited references for at least the reasons discussed above with reference to claim 1.

With respect to claims 10-33, claims 10-33 are rejected for reasoning analogous to claims 1-8. Thus, Applicant respectfully submits that claims 10-33 are likewise allowable over the cited references for at least the reasons discussed above.

With respect to claims 34-35, claims 34-35 depend from claim 32 and Applicant respectfully submits that claims 34-35 are likewise allowable over the cited references for at least the reasons discussed above with reference to claim 32.

With respect to claims 36-39, claims 36-39 are rejected for reasoning analogous to claims 32-35. Thus, Applicant respectfully submits that claims 36-39 are likewise allowable over the cited references for at least the reasons discussed above.

Claim 40 stands rejected under 35 U.S.C. §103(a) as being unpatentable over U.S. Patent No. 5,815,702 to Kannan et al. (hereinafter "Kannan") in view of U.S. Patent No. 5,826,082 to Bishop et al. (hereinafter "Bishop"). Applicant respectfully submits that claim 40 is not obvious over Kannan in view of Bishop.

Kannan discloses a system in which an exception handler intercepts exceptions that are fatal and that would normally cause an application to be terminated (see, col. 4, lines 44-47). The exception handler notifies a crash guard process of the fatal exception, which in turn interacts with the user to determine whether to continue executing the application or to terminate it (see, col. 4, lines 47-51). The crash guard process displays a warning dialog that allows the user to choose between continuing to work and terminating the application (see, Fig. 4, and col. 7, lines 34-47). Thus, the system of Kannan allows a user to continue using an application that has generated a fatal exception that would otherwise have caused the operating system to terminate execution of the application (see, col. 2, lines 39-43).

Bishop, on the other hand, discloses a computer system including a resource manager that is responsible for allocation of the computer system's resources (see, col. 2, lines 35-37). A thread that needs a resource submits a

request to the resource manager for the necessary resource including a requested amount of the resource (see, col. 3, lines 53-62). If the requested amount is not available, then the resource manager searches for a prior request(s) of a separate thread that has already reserved enough of the resource to satisfy the request, and attempts to suspend the prior request(s) (see, col. 4, lines 52-57 and 63-67; and col. 5, lines 1-4 and 23-31). This suspending of the prior request is transparent to the user of the computer system (see, col. 5, lines 30-31).

Applicant respectfully submits that it would not have been obvious to one of ordinary skill in the art to combine Kannan and Bishop. As discussed above, Kannan is directed to allowing a user to continue using an application that has generated a fatal exception that would otherwise have caused the operating system to terminate execution of the application, whereas Bishop is directed to reserving resources and suspending prior requests transparently to the user. Applicant respectfully submits that there is no teaching or suggestion to combine the allowing a user to select between continuing and terminating an application that has generated a fatal exception of Kannan with the resource management and user-transparent resource request suspension of Bishop.

However, assuming for the sake of argument that the Kannan and Bishop references are combined, Applicant respectfully submits that the combination does not disclose or suggest an application program that resides in a computer-readable memory for execution by a processor as recited in claim 40. Claim 40 recites:

> An application program that resides in a computer-readable memory for execution by a processor in conjunction with an operating system, the application program having a message loop that receives messages from an operating system, the application program being responsive to a particular message received through its message loop to reduce its current use of memory.

Kannan is not cited as disclosing, and Applicant respectfully submits that Kannan does not disclose or suggest, an application program being responsive to a particular message received through its message loop to reduce its current use of memory as claimed in claim 40.

Bishop, on the other hand, is cited as disclosing an application operation that is programmed to reduce its current use of memory (see, ¶5, p. 5). Applicant respectfully disagrees with this assertion. Bishop discloses a resource manager that is responsible for allocation of the computer's resources and that has the ability to suspend requests for resources (see, col. 5, lines 26-31). The resource manager determines whether to suspend a prior request of a thread based on the priorities of the current and prior requests, or the priorities of the current and prior requesting threads (see, col. 5, lines 17-22). Neither the current nor the prior requesting thread determines whether its request is to be suspended – the resource manager makes that determination. Thus, since neither of the threads makes this determination, neither of the threads itself is programmed (nor is the program the threads are part of programmed) to reduce its current use of memory (the resource manager forces one of their requests to be suspended, and thus forces a suspension in the use of resources by the thread whose request is suspended).

The cited portion of Bishop (col. 3, line 63-col. 4, line 5) discusses the beginning of the steps involved in reserving a resource. The cited portion of Bishop discloses determining if the requested amount of the requested resource is available (or if the amount available to be reserved is below a predetermined threshold). This determination is the basis for determining whether the resource manager needs to attempt to suspend a prior request (see, col. 4, lines 52-57). Applicant respectfully submits that nothing in this cited portion discloses or

suggests an application program being responsive to a particular message received through its message loop to reduce its current use of memory as claimed in claim 40.

Thus, Applicant respectfully submits that Bishop does not cure the deficiencies of Kannan.

For at least these reasons, Applicant respectfully submits that claim 40 is allowable over Kannan in view of Bishop.

Conclusion

Claims 1-40 are in condition for allowance. Applicant respectfully requests reconsideration and issuance of the subject application. Should any matter in this case remain unresolved, the undersigned attorney respectfully requests a telephone conference with the Examiner to resolve any such outstanding matter.

Respectfully Submitted,

Date: December 6, 2001

By: _____

Allan T. Sponseller
Reg. No. 38,318
(509) 324-9256